

# SD50 Process Data Function

---

February 3<sup>rd</sup>, 2025

This document covers the installation and use of a function for Siemen's TIA Portal software package. This function handles cyclic IO-Link Process Data Out to a Banner SD50 light via an IO-Link Master from Siemens PLC. The function covers parsing and display of the SD50 sensor Process Data Out.

## Components

Banner SD50 Library v16.zal16

There are two methods for process data. The first is used when creating a connection to Banner's IO-Link masters. The second set of instructions are for systems using other manufacturers' IO-Link masters.

### **Installation Instructions**

1. Open a project.
2. Go to the Open Global Library option in the Libraries tab in TIA Portal v16 or greater.



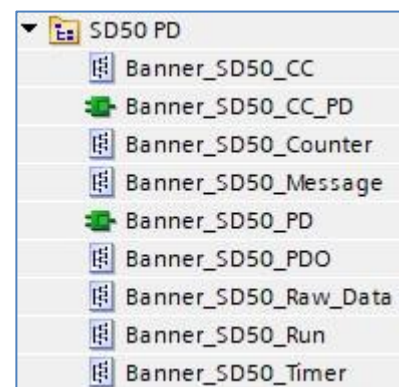
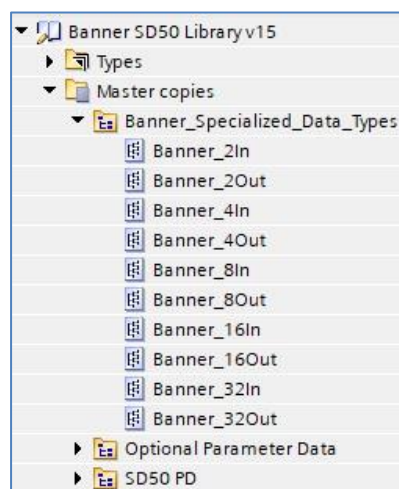
3. Switch the “Files of type” to Compressed libraries. Go to the location of the compressed library.
4. Press the Open button and the library will be uncompressed and opened.
5. The library is now accessible in the Libraries tab in v16 or greater.

**Setup of SD50 with a Banner DXMR**

1. Go to Device and Networks to configure the DXMR. Add the DXMR if it has yet to be added to the system.
2. Add Banner IO-Link Master Info to Slot 1. This sets the DXMR for IO-Link mode.
3. Open the IO-Link Generic Devices and select the proper module. The 32/32 byte is required for SD50 Select. Make note of the Q address for Slot 2 which represents Port 1 (%Q2).

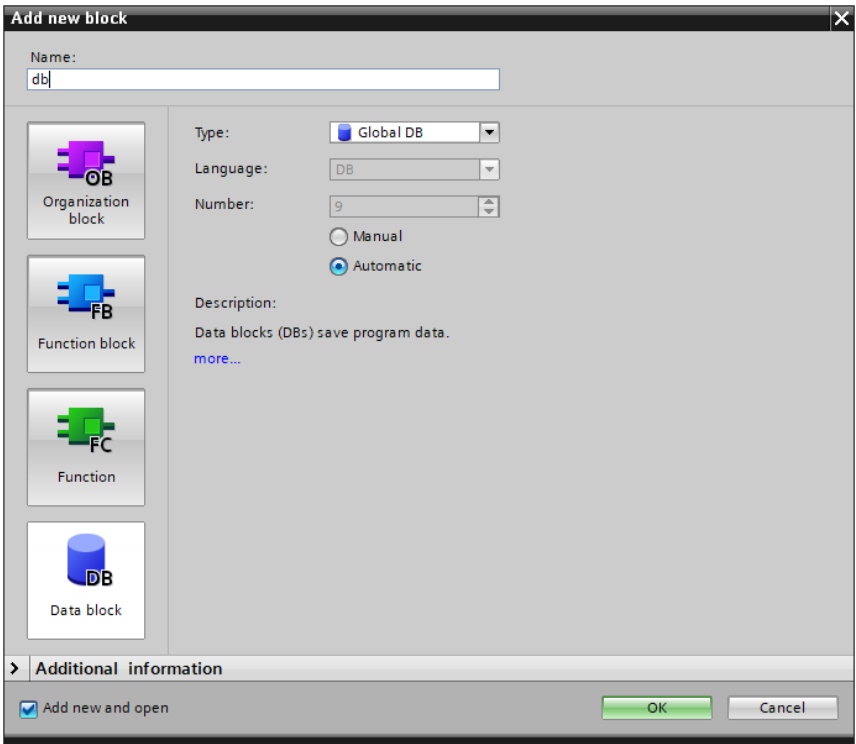
Module	Rack	Slot	I address	Q address	Type
▼ dxm	0	0			1-port Device
▶ Interface	0	0 X1			dxm
Banner IO-Link Master Info_1	0	1	68...76		Banner IO-Link Master Info
IO-Link In/Out 32/32 Byte + Status_1	0	2	2...37	2...47	IO-Link In/Out 32/32 Byte + Status

4. Drag the necessary tag from Banner\_Specialized\_Data\_Types. The tag used in this example is "Banner\_32Out". This tag represents the full raw process data along with port information.
5. Drag the necessary files from the SD50 PD Folder.
  - a. Move Banner\_SD50\_CC, Banner\_SD50\_Counter, Banner\_SD50\_Message, Banner\_SD50\_PDO, Banner\_SD50\_Raw\_Data, Banner\_SD50\_Run, and Banner\_SD50\_Timer to the PLC Data Types area.
  - b. Move Banner\_SD50\_CC\_PD and Banner\_SD50\_PD to the Program Blocks area.
6. Go to PLC Tags. Create two tags. One tag is for the full data structure while the second creates a tag to represent the raw Process Data from the IO-Link Master. In this example, Tag table\_1 was created, then the tag "SD50 IOLM1 01 PDO" was created using a Data Type of "Banner\_32Out". This naming convention calls out the type of device in question as well as the specific IO-Link Master and port number where the sensor is connected. A different IO-Link Master might be named IOLM2 or IOLM3, for instance, and other specific sensors may be connected to different port numbers. The "Q" address found in step 2 (%Q2) is tied to this new tag. The second is "SD50 IOLM1 01 outRaw" and uses the "Q" address found in step 3 plus 2 (%Q4). This is the tag that will be used in the Function block.



Name	Data type	Address
▶ SD50 IOLM1 01 PDO	"Banner_32Out"	%Q2.0
▶ SD50 IOLM1 01 outRaw	"Banner_SD50_Raw_Data"	%Q4.0

7. Go to Program blocks. Add a new Data block if necessary. In this example the new data block is named "db".



8. In the new data block, create a new tag to represent the parsed Process Data In for our SD50. The tag name again calls out the type of sensor, the IO-Link Master, and the port number. Use the data type “Banner\_SD50\_PDO” for the new tag.

Name	Data type
▼ Static	
■ ▼ SD50 IOLM1 01 PDO	*Banner_SD50_PDO*
■ ▶ 0-Run	*Banner_SD50_Run*
■ ▶ 1-Message	*Banner_SD50_Message*
■ ▶ 2-Measure	UInt
■ ▶ 3-Timer	*Banner_SD50_Timer*
■ ▶ 4-Counter	*Banner_SD50_Counter*

9. Add the “Banner\_SD50\_PD” function to an OB ladder. Link the “PDO” and “SD50 PDO” to the tags created from step 6.

The last variable, “Operational Mode”, allows the function to correctly interpret the Process Data Out. In the case of the SD50, there are five user-selected modes for the Process Data Out. This function needs to know what choice has been made in the SD50 for this Operational Mode variable.

There are two ways to achieve this goal. We can simply type in the correct number for Operational Mode (see Fig. 1), or we can link this SD50 Process Data Function to the SD50 Parameter Data Function Block (see Fig. 2). See Appendix A for more information about SD50 Process Data Out.

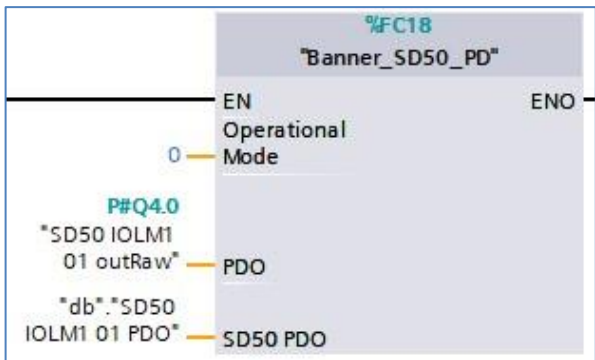


Figure 1: Hand type correct number for Operational Mode

**NOTE:** if you type in the incorrect number (i.e. it does not match the display module, SD50, current Operational Mode configuration) you will get incorrectly displayed Process Data Out information.

**Operational Mode:** the options here are “0” (Run Mode), “1” (Message Mode), “2” (Measure Mode), “3” (Timer Mode), and “4” (Counter Mode). The default is “0”.

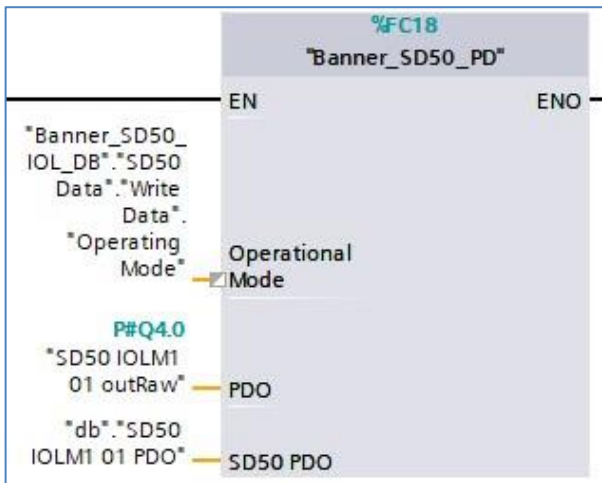
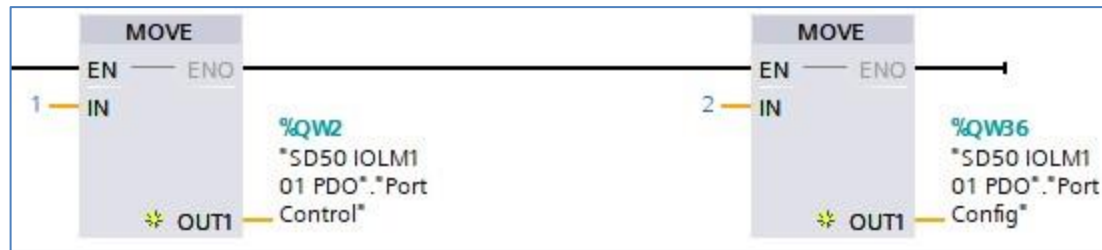


Figure 2: Linking Operational Mode variable to SD50 Parameter Data Function Block

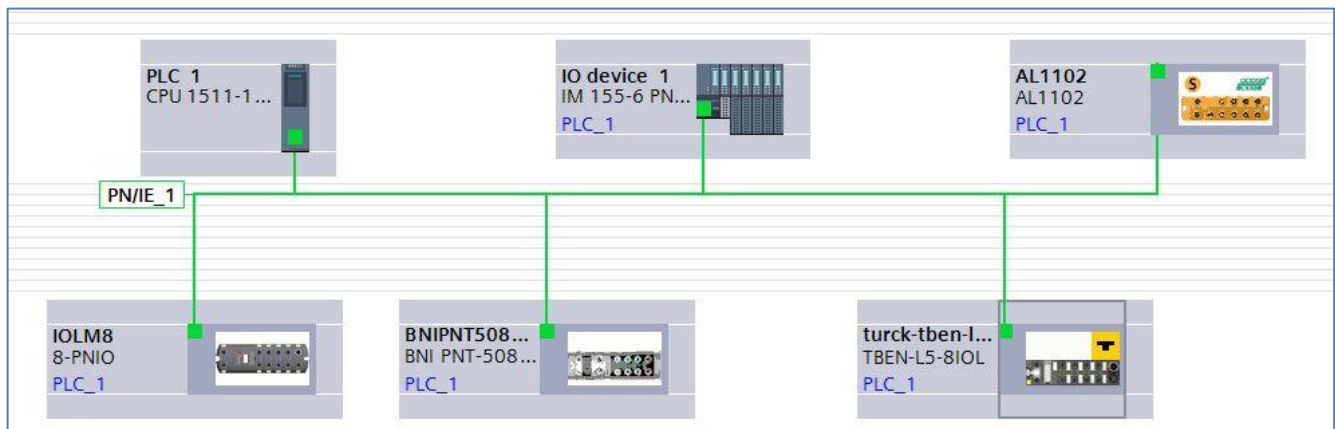
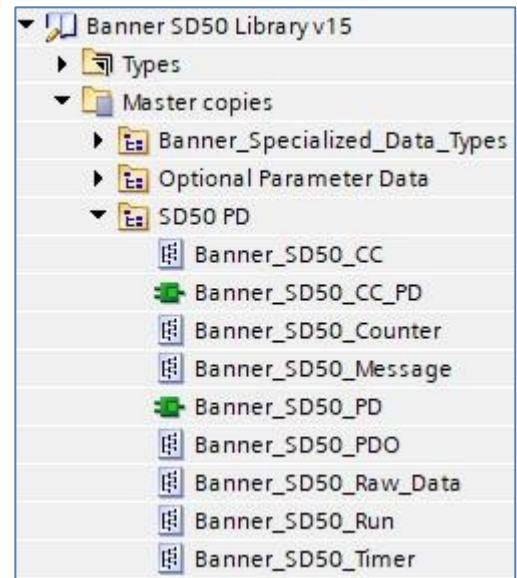
10. The final step is to configure the IO-Link output control. This is done by sending a 1 to Port Control and a 2 to Port Config. Both parameters are part of the tag created in step 6 "SD50 IOLM1 01 PDO".



11. Process Data setup is complete.
12. Compile and download the configuration to the PLC, then go online. Open the "db" data block and click Monitor all. You can now control the SD50 via the data located here.

### Setup of SD50 with other IO-Link Masters

1. The Banner SD50 Library will now be in the Global Library List. Expand the Master copies section. The SD50 folder contains elements for both Process Data and Parameter Data connections to a SD50 device. As Process Data is the focus of this paper on those items.
2. Drag Banner\_SD50\_CC\_PD and Banner\_SD50\_PD to the Program Blocks area under your PLC.
3. Drag Banner\_SD50\_CC, Banner\_SD50\_Counter, Banner\_SD50\_Message, Banner\_SD50\_PDO, Banner\_SD50\_Raw\_Data, Banner\_SD50\_Run, and Banner\_SD50\_Timer to the PLC Data Types area under your PLC.
4. Go to Devices and networks to configure the system as necessary. Below is an example of what a configuration might look like. This example shows 5 different IO-Link Masters connected to the same PLC.

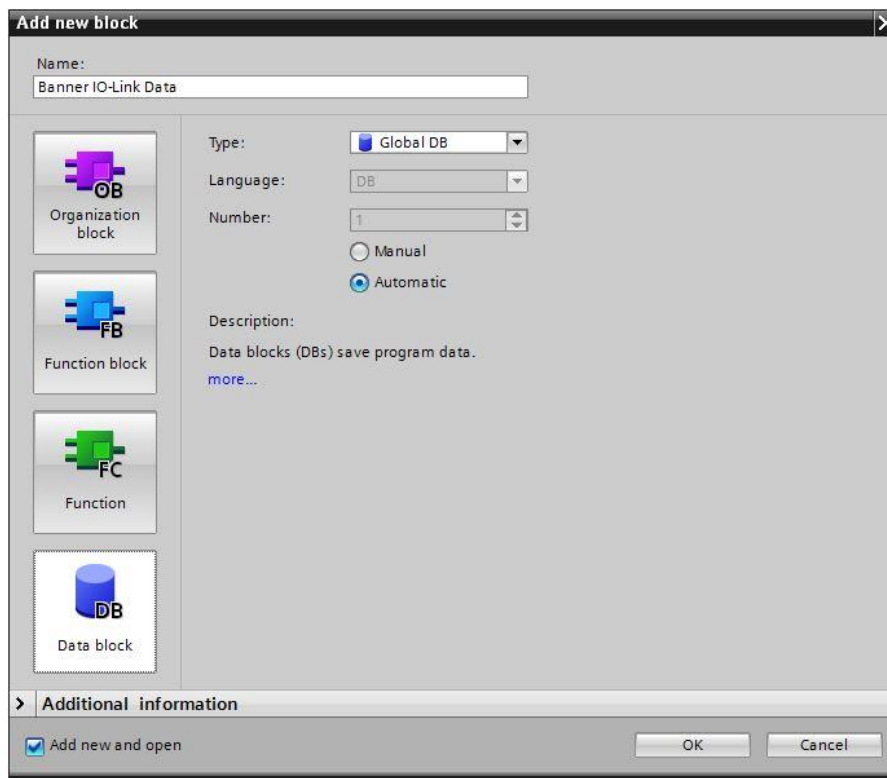


5. Click on the relevant device and configure the IO-Link Master as necessary. Refer to the documentation for the IO-Link Master. Recall that a SD50 requires 32 bytes of space for the Process Data.
6. Record the "Q" address where this SD50 Process Data is to be stored, as the address will be required in the next step. In this example, 32 bytes of Process Data Out for port 1 on the IO-Link Master will be stored in Q2 through Q33.

7. Go to PLC Tags. Add a new tag table, then create a new tag to represent the raw Process Data Out to be sent to the IO-Link Master. In this example, Tag table\_1 was created, then the tag "SD50 IOLM1 01 outRaw" was created using a Data Type of "Banner\_SD50\_Raw\_Data". This naming convention calls out the type of sensor in question as well as the specific IO-Link Master and port number where the sensor is connected. A different IO-Link Master might be named IOLM2 or IOLM3, for instance, and other specific sensors may be connected to different port numbers. Reference IO-Link Documentation for addressing. This example uses %Q2.

► SD50 IOLM1 01 outRaw "Banner\_SD50\_Raw\_Data" %Q2.0

8. Go to Program blocks. Add a new Data block if necessary. In this example the new data block is named "Banner IO-Link Data".





9. In the new data block, create a new tag to represent the parsed Process Data In for our SD50. The tag name again calls out the type of sensor, the IO-Link Master, and the port number. Use the data type "Banner\_SD50\_PDO" for the new tag.

Name	Data type
▼ Static	
■ ▼ SD50 IOLM1 01 PDO	"Banner_SD50_PDO"

Add the "Banner\_SD50\_PD" function to an OB ladder. Link the "PDO" to the raw Process Data variable from step 7. Link "SD50 PDO" to the parsed Process Data variable from step 9.

The last variable, "Operational Mode", allow the function to correctly interpret the Process Data Out. In the case of the SD50, there are four user-selected modes for the Process Data Out. This function needs to know what choice has been made in the SD50 for this Operational Mode variable.

There are two ways to achieve this goal. We can simply type in the correct number for Operational Mode (see Fig. 3), or we can link this SD50 Process Data Function to the SD50 Parameter Data Function Block (see Fig. 4). See Appendix A for more information about SD50 Process Data Out.

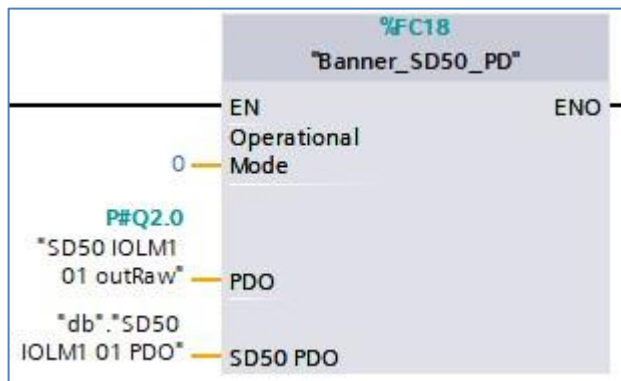


Figure 3: Hand type correct number for Operational Mode

**NOTE:** if you type in the incorrect number (i.e. it does not match the display module, SD50, current Operational Mode configuration) you will get incorrectly displayed Process Data Out information.

**Operational Mode:** the options here are "0" (Run Mode), "1" (Message Mode), "2" (Measure Mode), "3" (Timer Mode), and "4" (Counter Mode). The default is "0".

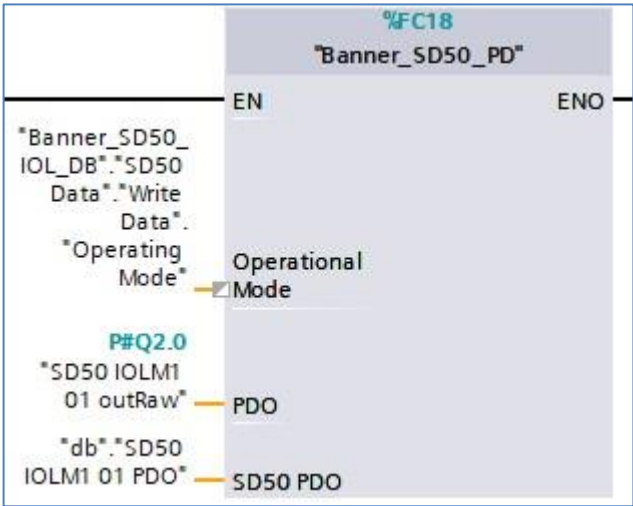


Figure 4: Linking Operational Mode variable to SD50 Parameter Data Function Block

- 10. Process Data setup is complete.
- 11. Compile and download the configuration to the PLC, then go online. Open the “Banner IO-Link Data” data block and click Monitor all. You should see parsed SD50 Process Data Out, like that shown below.

Name	Data type	Start value	Monitor value
▼ Static			
■ ▼ SD50 IOLM1 01 PDO	"Banner_SD50_PDO"		
■ ▼ 0-Run	"Banner_SD50_Run"		
■ ▼ Light Control	"Banner_SD50_CC"		
■ Animation	USInt	0	1
■ Color 1	USInt	0	7
■ Color 1 Inte...	USInt	0	0
■ Speed	USInt	0	0
■ Pulse Pattern	USInt	0	0
■ Color 2	USInt	0	0
■ Color 2 Inte...	USInt	0	0
■ Display Text	String	"	"Banner Light Bar"
■ ▶ 1-Message	"Banner_SD50_Message"		
■ ▶ 2-Measure	UInt	0	0
■ ▶ 3-Timer	"Banner_SD50_Timer"		
■ ▶ 4-Counter	"Banner_SD50_Counter"		

Figure 5: Run Mode Process Data Out (Operational Mode = 0)

Name	Data type	Start value	Monitor value
▼ Static			
■ ▼ SD50 IOLM1 01 PDO	"Banner_SD50_PDO"		
■ ▶ 0-Run	"Banner_SD50_Run"		
■ ▼ 1-Message	"Banner_SD50_Message"		
■ ▼ Light Control	"Banner_SD50_CC"		
■ Animation	USInt	0	1
■ Color 1	USInt	0	3
■ Color 1 Intensity	USInt	0	0
■ Speed	USInt	0	0
■ Pulse Pattern	USInt	0	0
■ Color 2	USInt	0	0
■ Color 2 Intensity	USInt	0	0
■ Message 1	USInt	0	0
■ Message 2	USInt	0	0
■ 2-Measure	UInt	0	0
■ ▶ 3-Timer	"Banner_SD50_Timer"		
■ ▶ 4-Counter	"Banner_SD50_Counter"		

Figure 6: Message Mode Process Data Out (Operational Mode = 1)

Name	Data type	Start value	Monitor value
▼ Static			
■ ▼ SD50 IOLM1 01 PDO	"Banner_SD50_PDO"		
■ ▶ 0-Run	"Banner_SD50_Run"		
■ ▶ 1-Message	"Banner_SD50_Message"		
■ 2-Measure	UInt	0	0

Figure 7: Measure Mode Process Data Out (Operational Mode = 2)

Name	Data type	Start value	Monitor value
▼ Static			
■ ▼ SD50 IOLM1 01 PDO	"Banner_SD50_PDO"		
■ ▶ 0-Run	"Banner_SD50_Run"		
■ ▶ 1-Message	"Banner_SD50_Message"		
■ 2-Measure	UInt	0	0
■ ▼ 3-Timer	"Banner_SD50_Timer"		
■ Run Timer	USInt	0	1
■ Reset Timer	USInt	0	0
■ ▶ 4-Counter	"Banner_SD50_Counter"		

Figure 8: Timer Mode Process Data Out (Operational Mode = 3)

Name	Data type	Start value	Monitor value
▼ Static			
■ ▼ SD50 IOLM1 01 PDO	"Banner_SD50_PDO"		
■ ▶ 0-Run	"Banner_SD50_Run"		
■ ▶ 1-Message	"Banner_SD50_Message"		
■ 2-Measure	UInt	0	0
■ ▼ 3-Timer	"Banner_SD50_Timer"		
■ Run Timer	USInt	0	1
■ Reset Timer	USInt	0	0
■ ▶ 4-Counter	"Banner_SD50_Counter"		

Figure 9: Counter Mode Process Data Out (Operational Mode = 4)

**Appendix A****SD50 Process Data Out**

The SD50 has 32 bytes of Process Data Out, mapped into 4 different modes, as shown below.

This Process Data is mapped to a specific group of PROFINET addresses. The 256-bits of Process Data encode many separate pieces of information, as shown below.

This function intelligently parses this Process Data into its component pieces.

First is Run mode (mode 0).

**ProcessDataOut "Process Data Out Run Mode" id=V\_Pd\_OutRunMode**

bit length: 256

data type: 256-bit Record

subindex	bit offset	data type	allowed values	default value	acc. restr.	mod. other var.	excl. from DS	name	description
1	0	4-bit UInteger	0 = Off, 1 = Steady, 2 = Flash, 3 = Two Color Flash, 4 = Half/Half Steady, 5 = Half/Half Flash, 6 = Intensity Sweep, 7 = Two Color Sweep					Animation	The Animation type
2	4	4-bit UInteger	0 = Green, 1 = Red, 2 = Orange, 3 = Amber, 4 = Yellow, 5 = Lime Green, 6 = Spring Green, 7 = Cyan, 8 = Sky Blue, 9 = Blue, 10 = Violet, 11 = Magenta, 12 = Rose, 13 = White, 14 = Custom 1, 15 = Custom 2					Color 1	The main color of the Animation. Custom Colors are defined in Parameter data
3	8	3-bit UInteger	0 = High, 1 = Medium, 2 = Low, 3 = Off, 4 = Custom					Color 1 Intensity	The Intensity of Color 1, Custom Intensity defined in Parameter Data
4	11	2-bit UInteger	0 = Slow, 1 = Standard, 2 = Fast, 3 = Custom					Speed	The speed of the Animation
5	13	3-bit UInteger	0 = Normal, 1 = Strobe, 2 = Three Pulse, 3 = SOS, 4 = Random					Pulse Pattern	The pattern of Animation
6	16	4-bit UInteger	0 = Green, 1 = Red, 2 = Orange, 3 = Amber, 4 = Yellow, 5 = Lime Green, 6 = Spring Green, 7 = Cyan, 8 = Sky Blue, 9 = Blue, 10 = Violet, 11 = Magenta, 12 = Rose, 13 = White, 14 = Custom 1, 15 = Custom 2					Color 2	The secondary color of the Animation. Only used if Animation has two colors. Custom Colors are defined in Parameter data
7	20	3-bit UInteger	0 = High, 1 = Medium, 2 = Low, 3 = Off, 4 = Custom					Color 2 Intensity	The Intensity of Color 2, Custom Intensity defined in Parameter Data
8	24	29-octet String UTF-8						Display Text	Run Mode Display Text

Here is the information for Measure mode (mode 1).

### ProcessDataOut "Process Data Out Message Mode" id=V\_Pd\_OutMessageMode

bit length: 256

data type: 256-bit Record

subindex	bit offset	data type	allowed values	default value	acc. restr.	mod. other var.	excl. from DS	name	description
1	224	4-bit UInteger	0 = Off, 1 = Steady, 2 = Flash, 3 = Two Color Flash, 4 = Half/Half Steady, 5 = Half/Half Flash, 6 = Intensity Sweep, 7 = Two Color Sweep					Animation	The Animation type
2	228	4-bit UInteger	0 = Green, 1 = Red, 2 = Orange, 3 = Amber, 4 = Yellow, 5 = Lime Green, 6 = Spring Green, 7 = Cyan, 8 = Sky Blue, 9 = Blue, 10 = Violet, 11 = Magenta, 12 = Rose, 13 = White, 14 = Custom 1, 15 = Custom 2					Color 1	The main color of the Animation. Custom Colors are defined in Parameter data
3	232	3-bit UInteger	0 = High, 1 = Medium, 2 = Low, 3 = Off, 4 = Custom					Color 1 Intensity	The Intensity of Color 1, Custom Intensity defined in Parameter Data
4	235	2-bit UInteger	0 = Slow, 1 = Standard, 2 = Fast, 3 = Custom					Speed	The speed of the Animation
5	237	3-bit UInteger	0 = Normal, 1 = Strobe, 2 = Three Pulse, 3 = SOS, 4 = Random					Pulse Pattern	The pattern of Animation
6	240	4-bit UInteger	0 = Green, 1 = Red, 2 = Orange, 3 = Amber, 4 = Yellow, 5 = Lime Green, 6 = Spring Green, 7 = Cyan, 8 = Sky Blue, 9 = Blue, 10 = Violet, 11 = Magenta, 12 = Rose, 13 = White, 14 = Custom 1, 15 = Custom 2					Color 2	The secondary color of the Animation. Only used if Animation has two colors. Custom Colors are defined in Parameter data
7	244	3-bit UInteger	0 = High, 1 = Medium, 2 = Low, 3 = Off, 4 = Custom					Color 2 Intensity	The Intensity of Color 2, Custom Intensity defined in Parameter Data
8	248	4-bit UInteger	1..13 = Message Selection (1-13)					Message Selection 1	Message Mode Message Selection 1 (1-13)
9	252	4-bit UInteger	1..13 = Message Selection (1-13)					Message Selection 2	Message Mode Message Selection 2 (1-13)

Here is Measure mode (mode 2).

ProcessDataOut "Process Data Out Measure Mode" id=V_Pd_OutMeasureMode									
bit length: 256 data type: 256-bit Record									
subindex	bit offset	data type	allowed values	default value	acc. restr.	mod. other var.	excl. from DS	name	description
1	240	16-bit UInteger						Measure Mode Value	Value describing the level of the device, range determined in Measure Mode Parameter Data

Here is Timer mode (mode 3).

ProcessDataOut "Process Data Out Timer Mode" id=V_Pd_OutTimerMode									
bit length: 256 data type: 256-bit Record									
subindex	bit offset	data type	allowed values	default value	acc. restr.	mod. other var.	excl. from DS	name	description
1	254	Boolean						Run Timer	Run the timer in timer mode
2	255	Boolean						Reset Timer	Reset the timer in timer mode

Here is Counter mode (mode 4).

ProcessDataOut "Process Data Out Counter Mode" id=V_Pd_OutCounterMode									
bit length: 256 data type: 256-bit Record									
subindex	bit offset	data type	allowed values	default value	acc. restr.	mod. other var.	excl. from DS	name	description
1	253	Boolean						Increment Count	Increment the counter value when in counter mode
2	254	Boolean						Decrement Count	Decrement the counter value when in counter mode
3	255	Boolean						Reset Count	Reset the count in counter mode